REMARKS

The Examiner rejected claims 1-20 under 35 U.S.C. §102(e) as allegedly being anticipated by Mauricio Bretemitz, Jr. et al. (U.S. Patent No. 6,381,739).

Applicants respectfully traverse the §102(e) rejections with the following arguments, which reflect a telephone discussion between the Examiner and Applicant's representative on 04/22/2005.

35 U.S.C. §102(e)

For clarification, Applicant notes that the description of prior art in the BACKGROUND OF THE INVENTION section (i.e., FIGS. 1-5 and col. 1, line 25 - col. 4, line 43) in Breternitz is denoted herein as "Breternitz Prior Art", whereas the description of Breternitz' invention in the DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT and Claims sections (i.e., col. 6, line 14 - col. 40, line 10) in Breternitz is denoted herein as "Breternitz Invention".

Claims 1-6

The Examiner rejected claims 1-20 under 35 U.S.C. §102(e) as allegedly being anticipated by Mauricio Breternitz, Jr. et al.

As an introduction to Applicants's arguments traversing the rejection of claim 1, Applicants next explain claim 1 as applied to an embodiment of the present invention described in pages 9-10 of Applicant's specification. The following discussion will also explain some of the terminology used in claim 1.

The procedure "getsomevalue" begins in the specification on page 9, line 3 and comprises control flow code in the form of an "if ... else ..." construct, which includes a branch condition of "p==NULL" and two code branches. The first code branch includes "ch = 0; return ch", and the second code branch includes "ELSE_BODY; return CH". The procedure "getsomevalue" is invoked by the call statement "Val = getsomeval(" on page 9, line 25. Step (a) of claim 1 identifies the aforementioned control flow code and code branches.

Step (b) of claim 1 identifies for each of the preceding code branches a new procedure containing the respective code branch as shown on page 10 of the specification. In particular, page 10 of the specification shows the first new procedure "getsomevalue_0" which contains the first code branch "return 0", and also shows the second new procedure "getsomevalue_1" which contains the second code branch "ELSEBODY; Return ch".

Steps (c) and (d) of claim 1 are described in detail on pages 12-14 of the specification in terms of recording a scanning a list of data entries, called PROCEDURE_CLONE_LIST(P), for each new procedure.

Applying step (c) of claim 1 to the example of pages 9-10, the list of data entries for the first new procedure "getsomevalue_0" comprises a first data entry identifying the first new procedure "getsomevalue_0" and a second data entry identifying the branch condition of "p = = NULL" which directs program flow to the first code branch "return 0". In addition, the list of data entries for the second new procedure "getsomevalue_1" comprises a first data entry identifying the second new procedure "getsomevalue_1" and a second data entry identifying the branch condition of "p ≠ NULL" (i.e., the "else" component of the "if ... else ..." construct) which directs program flow to the second code branch "ELSEBODY; Return ch".

Step (d) of claim 1 scans the list of data entries recorded in step (c), and makes use of the recorded data entries to replace the call of "val=getsomevalue(" to the procedure "getsomevalue" on page 9 with the two calls of "val=getsomevalue_0" and "val=getsomevalue_1" inside a control flow code that utilizes the first branch condition (of "p = = NULL") and the second branch condition (of p ≠ NULL as implied by "else") as depicted on the bottom of page 10 of the specification..

In summary, step (b) of claim 1 identifies new procedures such that each new procedure contains a different code branch of the original procedure. Step (d) replaces the call to the original procedure by calls to the new procedures by a construct that is generated from a list of data entries in step (c).

Applicant next explains that neither Breternitz does not teach each and every feature of claim 1. For example, Breternitz does not teach step (b) of claim 1, wherein step (b) recites: "identifying for each said code branch a new **procedure** containing the respective code branch" (emphasis added). In particular, Breternitz does not teach each said "new procedure"

FIG. 1 of Breternitz Prior Art is a control flow graph such that the nodes a, b, ..., j are procedures and the edges show program flow from one procedure to another (e.g., the edge a →b denotes that execution of procedure a is immediately followed by execution of procedure b). The only procedures in FIG. 1 containing multiple code branches are procedures b and f. Using procedure b as an example, the code branches of procedure b are procedures c and d, wherein a directing of program flow to code branches c and d from procedure a are implemented by an if-then-else instruction or the like (see Breternitz, col. 1, lines 45-50). However, Breternitz Prior

Art does not identify two new procedures each containing the code branches c and d, as required by step (b) of claim 1. Instead, Bretemitz Prior Art teaches using trace data (see FIG. 2 of Bretemitz) empirically obtained from multiple executions of the program of FIG. 1 of Breternitz to compute edge weights as shown in FIG. 3 of Bretemitz. Bretemitz Prior Art uses the edge weights to derive a path (see FIG. 5 of Bretemitz) from procedure a to procedure j such that the derived path has a high probability of being traversed in comparison with alternative paths from procedure a to procedure j. The purpose of deriving the path of FIG. 5 of Bretemitz is to "to determine an efficient manner of ordering basic blocks in memory so that cache performance may be improved and pipeline flushing may be minimized resulting in improved processor performance" as recited in Bretemitz, col. 2, line 66 - col. 3, line 4.

Thus, Bretemitz Prior Art teaches an ordering of blocks to memory to effectuate improved execution speed through a more efficient cache performance. However, Bretemitz Prior Art does not teach identifying new procedures each containing the respective code branches of c and d that are contained in the procedure b, as required in step (b) of claim 1.

In addition, Bretemitz Prior Art does not teach step (c) of claim 1 which recites: "recording a list of data entries corresponding to the respective new procedures, each entry comprising a data item identifying the respective new procedure and a data item representative of the branch conditions under which said control flow code directs program flow to the associated code branch". The Examiner alleges that the trace data of Bretemitz, FIG. 2 represents the "list of data entries" of step (c). However, the trace data of FIG. 2 do not identify the respective new procedures of step (b).

In addition, Bretemitz Prior Art does not teach step (d) of claim 1 which recites: "for the or each call statement, scanning the entries in said list to determine one for which there is correspondence between said branch conditions and call parameters directed to said control flow code by the call statement and modifying the call statement to replace the call to the original procedure by a call to the corresponding new procedure". Applicant maintains that there is no disclosure in Bretemitz Prior Art of modifing the original call statement with replacement call statements that call the new procedures respectively containing the code branches c and d. In addition, Bretemitz Prior Art does not teach scanning the entries in the list of data entries as claimed.

Breternitz Invention solves the same problem as Breternitz Prior Art but does so more accurately. Col. 4, lines 15-36 of Breternitz identifies a logical flaw in Breternitz Prior Art due to failure to account for coupling between sequentially separated edges, resulting in a non-optimal path in FIG. 5 of Breternitz. Breternitz Invention discloses a more sophisticated method of analyzing trace data (e.g., see Breternitz, FIG. 9) to account for weighted edges between nodes that are not directcly coupled by program flow, as illustrated by the dotted lines in FIG. 10 of Breternitz and explained in Breternitz, col. 9, lines 44-50. However, as in Breternitz Prior Art, Applicant respectfully contends that Breternitz Invention does not disclose steps (b), (c), and (d) of claim 1 for any of the graphs of Breternitz Invention (i.e., FIGS. 10, 14-27, 34-35, 37, and 43), for the same arguments described *supra* in relation to Breternitz Prior Art.

Based on the preceding arguments, Applicants respectfully maintain that Breternitz does not anticipate claim 1, and that claim 1 is in condition for allowance. Since claims 2-6 depend from claim 1, Applicants contend that claims 2-6 are likewise in condition for allowance.

Claim 7

Applicants respectfully contend that Breternitz does not anticipate claim 7, because Breternitz does not teach each and every feature of claim 7.

As a first example of why Breternitz Prior Art does not anticipate claim 7, Breternitz does not teach the feature in step (b): "identifying a new **procedure** for which the flow control graph comprises all the nodes in the path from the first node of the procedure to the node being considered, the node being considered, and the whole of the portion of the control flow graph led to directly or indirectly from the node being considered" (emphasis added), based on the same arguments presented *supra* in relation to step (b) of claim 1.

As a second example of why Breternitz Prior Art does not anticipate claim 7, Breternitz does not teach the feature of step (c): "recording a list of data entries corresponding to the respective new procedures, each entry comprising a data item identifying the respective new procedure and a data item representative of the corresponding branching condition'", based on the same arguments presented *supra* in relation to step (c) of claim 1.

As a third example of why Breternitz Prior Art does not anticipate claim 7, Breternitz does not teach the features of steps (d) and (e): "for each said call statement, scanning the entries in said list to determine one for which there is correspondence between said branch condition and call parameters supplied by the call statement; and ... modifying the call statements to call said new procedures", based on the same arguments presented *supra* in relation to step (d) of claim 1. In claim 7.

Based on the preceding arguments, Applicants respectfully maintain that Breternitz Prior Art does not anticipate claim 7, and that claim 7 is in condition for allowance.

Claims 8-13

Applicants respectfully contend that Breternitz does not anticipate claim 8, because Breternitz does not teach each and every feature of claim 8.

As a first example of why Breternitz Prior Art does not anticipate claim 8, Breternitz does not teach the feature: "identifying means for identifying each said code branch **a new procedure** containing the respective code branch" (emphasis added), based on the same arguments presented *supra* in relation to step (b) of claim 1.
the preceding feature of claim 8.

As a second example of why Breternitz Prior Art does not anticipate claim 8, Breternitz does not teach the feature: "recording means for recording a list of data entries corresponding to the respective new procedures, each entry comprising a data item identifying the respective new procedure and a data item representative of the branch conditions under which said control flow code directs program flow to the associated code branch", based on the same arguments presented *supra* in relation to step (c) of claim 1.

As a third example of why Breternitz Prior Art does not anticipate claim 8, Breternitz does not teach the feature: "scanning means operable, for each said call statement, for scanning the entries in said list to determine one for which there is correspondence between said branch conditions and call parameters supplied by the call statement; and ... modifying means for modifying the call statement to call the corresponding new procedure", based on the same arguments presented *supra* in relation to step (d) of claim 1.

Based on the preceding arguments, Applicants respectfully maintain that Breternitz does not anticipate claim 8, and that claim 8 is in condition for allowance. Since claims 9-13 depend from claim 8, Applicants contend that claims 9-13 are likewise in condition for allowance.

Claim 14

Applicants respectfully contend that Breternitz does not anticipate claim 14, because Breternitz does not teach each and every feature of claim 14.

As a first example of why Breternitz Prior Art does not anticipate claim 14, Breternitz does not teach the feature: "identifying **a new procedure** for which the flow control graph comprises all the nodes in the path from the first node of the procedure to the node being considered, the node being considered, and the whole of the portion of the control flow graph led to directly or indirectly from the node being considered" (emphasis added), based on the same arguments presented *supra* in relation to step (b) of claim 1.

As a second example of why Breternitz Prior Art does not anticipate claim 14, Breternitz does not teach the feature: "means for recording a list of data entries corresponding to the respective new procedures, each entry comprising a data item identifying the respective new procedure and a data item representative of the corresponding branching condition", based on the same arguments presented *supra* in relation to step (c) of claim 1.

As a third example of why Breternitz Prior Art does not anticipate claim 14, Breternitz does not teach the feature: "means for scanning the entries in said list to determine for each call statement, an entry for which there is correspondence between said branch condition and call parameters supplied by the call statement; and ... means for modifying the call statements to call said new procedures", based on the same arguments presented *supra* in relation to step (d) of claim 1.

Based on the preceding arguments, Applicants respectfully maintain that Breternitz does not anticipate claim 14, and that claim 14 is in condition for allowance.

Claim 15

Applicants respectfully contend that Bretemitz does not anticipate claim 15, because Bretemitz does not teach each and every feature of claim 15.

As a first example of why Bretemitz Prior Art does not anticipate claim 15, Bretemitz does not teach the feature: "second computer code portion for identifying for each said code branch a **new procedure** containing the respective code branch" (emphasis added), based on the same arguments presented *supra* in relation to step (b) of claim 1.

As a second example of why Bretemitz Prior Art does not anticipate claim 15, Bretemitz does not teach the feature: "a third computer code portion for recording a list of data entries corresponding to the respective new procedures, each entry comprising a data item identifying the respective new procedure and a data item representative of the branch conditions under which said control flow code directs program flow to the associated code branch", based on the same arguments presented *supra* in relation to step (c) of claim 1.

As a third example of why Bretemitz Prior Art does not anticipate claim 15, Bretemitz does not teach the feature: "a fourth computer code portion operable, for each said call statement, for scanning the entries in said list to determine one for which there is correspondence between said branch conditions and call parameters supplied by the call statement; and ... a fifth computer code portion for modifying the call statement to call the corresponding new procedure", based on the same arguments presented *supra* in relation to step (d) of claim 1.

Based on the preceding arguments, Applicants respectfully maintain that Bretemitz does not anticipate claim 15, and that claim 15 is in condition for allowance. Since claims 16-17 depend from claim 15, Applicants contend that claims 16-17 are likewise in condition for allowance.

Claim 18

Applicants respectfully contend that Bretemitz does not anticipate claim 18, because Bretemitz does not teach each and every feature of claim 18.

As a first example of why Bretemitz Prior Art does not anticipate claim 18, Bretemitz does not teach the feature: "identifying a new procedure for which the flow control graph comprises all the nodes in the path from the first node of the procedure to the node being considered, the node being considered, and the whole of the portion of the control flow graph led to directly or indirectly from the node being considered" (emphasis added), based on the same arguments presented *supra* in relation to step (b) of claim 1.

As a second example of why Breternitz Prior Art does not anticipate claim 18, Breternitz does not teach the feature: "a third code portion for recording a list of data entries corresponding to the respective new procedures, each entry comprising a data item identifying the respective new procedure and a data item representative of the corresponding branching condition", based on the same arguments presented *supra* in relation to step (c) of claim 1.

As a third example of why Breternitz Prior Art does not anticipate claim 18, Breternitz does not teach the feature: "a fourth code portion operable, for each said call statement, for scanning the entries in said list to determine one for which there is correspondence between said branch condition and call parameters supplied by the call statement; and ... a fifth code portion for modifying the call statements to call said new procedures", based on the same arguments presented *supra* in relation to step (d) of claim 1.

Based on the preceding arguments, Applicants respectfully maintain that Breternitz does not anticipate claim 18, and that claim 18 is in condition for allowance. Since claims 19-20 depend from claim 18, Applicants contend that claims 19-20 are likewise in condition for allowance.

CONCLUSION

Based on the preceding arguments, Applicants respectfully believe that all pending claims and the entire application meet the acceptance criteria for allowance and therefore request favorable action. If the Examiner believes that anything further would be helpful to place the application in better condition for allowance, Applicants invites the Examiner to contact Applicants' representative at the telephone number listed below. The Director is hereby authorized to charge and/or credit Deposit Account No. 09-0457.

Date: 04/26/2005

Jack P. Friedman
Jack P. Friedman
Registration No. 44,688

Schmeiser, Olsen & Watts
3 Lear Jet Lane, Suite 201
Latham, New York 12110
(518) 220-1850